Defense Threat Reduction Agency
8725 John J. Kingman Road, MS-6201
Fort Belvoir, VA 22060-6201

DTRA-TR-19-039

TECHNICAL REPORT

Design Aspects of DTRA's NuTRIS-Web Software Application

March 2020

HDTRA1-15-C-0002

Prepared by:
CACI, Inc.
11955 Freedom Drive, Suite 12000
Reston, VA 20190-5687

REPORT DOCUMENTATION PAGE	*Form Approved* **OMB No. 0704-0188**

1. REPORT DATE *(DD-MM-YYYY)* 10-03-2020	**2. REPORT TYPE** Technical report	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE Design Aspects of DTRA's NuTRIS-Web Software Application	**5a. CONTRACT NUMBER** HDTRA1-15-C-0002
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Siruvuri, Raju; Giannelli, Priscilla	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) CACI, Inc. 11955 Freedom Drive, Suite 12000 Reston, VA 20190-5687	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Nuclear Technologies Department, Attn: LCDR Franks, USN Defense Threat Reduction Agency 8725 John J. Kingman Road, Mail Stop 6201 Fort Belvoir, VA 22060-6201	**10. SPONSOR/MONITOR'S ACRONYM(S)** DTRA RD-NTS
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER DTRA-TR-19-039

12. DISTRIBUTION AVAILABILITY STATEMENT
DISTRIBUTION A. Approved for public release: distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
DTRA's new Nuclear Test Review and Information System web application (NuTRIS-Web) provides expanded functionality for the Nuclear Test Personnel Review (NTPR) program's repository of atomic veteran information in support of veteran verification and associated radiation dose assessment. This document provides a comprehensive architectural and system design overview of NuTRIS-Web. It also provides an overview of the technologies used to develop and implement the application.

15. SUBJECT TERMS
NuTRIS-Web, Atomic Veterans, Solution Architecture, Technical Design, Data Model

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** LCDR James D. Franks, USN
a. REPORT U	**b.** ABSTRACT U	**c.** THIS PAGE U	U	29	**19b. TELEPONE NUMBER (*Include area code*)** *571.616.6015*

STANDARD FORM 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Table of Contents

List of Figures

List of Tables

Acknowledgements

The authors gratefully acknowledge the superb peer review of this document by Aaron Parks of Applied Research Associates, Inc., Raleigh, NC and programmatic support by Dr. Paul Blake of DTRA’s Nuclear Survivability Division. This work was performed under DTRA contract HDTRA1-15-C-0002.

Section 1. Introduction

This report discusses the technical aspects of the Defense Threat Reduction Agency's (DTRA) Nuclear Test Review and Information System (NuTRIS) Web application. The NuTRIS system was built to fulfill one of the Nuclear Test Personnel Review (NTPR) Program's objectives to establish and maintain a comprehensive and accessible repository of DoD personnel involved in the U.S. atmospheric nuclear tests in addition to including historical and radiological information. The NTPR Program is now responsible for identifying DoD personnel as participants in five missions:

- participants of U.S. atmospheric nuclear tests (1945 - 1962), conducted primarily in Nevada and the Pacific Ocean;
- post-World War II occupation forces of Hiroshima and Nagasaki;
- prisoners of war in Japan at the conclusion of World War II;
- participants of U.S. underground nuclear testing between (1951 - 1992), conducted primarily in Nevada; and
- participants of the radiological cleanup of the Bikini and Enewetak Atolls.

As of February 2020, there are approximately 551,462 personnel records contained in NuTRIS-Web, 492,245 of which are confirmed participants (59,217 non-participants).
The NTPR Program uses NuTRIS-Web to record participant data including radiation dose information and to perform case management activities tracking the veteran's claim for filing radiogenic disease compensation claims with the Department of Veterans Affairs, Department of Justice, and the Department of Labor.

Additional features in NuTRIS-Web include an electronic paperless case file workflow, the ability to track the status of any case file, links to veteran digitized personnel files, and case file metrics that track the number of open cases, cases in progress, and unassigned cases. Authorized users access NuTRIS-Web via a PC with internet connection and a DoD Common Access Card (CAC) into the DTRA unclassified network (UNET) domain.

Section 2. Purpose

This document describes the high-level system design and architecture using several different architectural views to depict different parts of the system. This document will be used to guide the iterative development of NuTRIS-Web.

Specifically, this document addresses the following questions:

- What are the architecture goals and constraints?
- What is the technical architecture for the system?
- What are the design and guiding principles?
- What are the design considerations?

Section 3. Background

NTPR Database
The original version of the NuTRIS system, known as the NTPR database, was used between 1983 and 1999 and supported by a Digital Equipment Corporation VAX/VMS mainframe operating system. The database contained participant and dose information for approximately 400,000 records of military and DoD civilians.

NuTRIS Database
In 1994, the NTPR Program initiated the development of a client-server system. This initiative would improve the NTPR Database functionality and user access to program data. The initial version of the new NuTRIS database, developed by Diverse Technologies Corporation (DTC), was delivered in October 15, 1997. JAYCOR, the support contractor, performed extensive testing and a functional assessment of the system. JAYCOR concluded the system did not provide all the functionality required to support the program. Furthermore, they found that the database structure could not effectively support the requirements and would need to be redesigned to increase the efficiency of the application, remove redundant data, and provide a more flexible approach to access data and incorporate future changes. Based on this feedback, JAYCOR was selected to undertake the effort to fully redesign the database structure, the functionality of the system, and the user interface.

NuTRIS PowerBuilder Application
In March of 1999, JAYCOR delivered version 1.0 of the NuTRIS PowerBuilder application. The new application included the complete functionality of the legacy NTPR system, a redesigned database, and access to the NuTRIS application from users' desktops. The NuTRIS database was built using a PowerBuilder front-end and an MS SQL Server database that ran on a Microsoft Windows-based platform. The application currently uses PowerBuilder 12.6 and Microsoft SQL Server 2008 R2 and has approximately 48 screens, 45 reports and queries, and over 90 database tables.

NuTRIS-Web
In 2018, NTPR focused on modernizing the legacy NuTRIS PowerBuilder application to a web-based application using Microsoft .NET web technologies. On August 28, 2019, NuTRIS-Web version 1.0.0 went live. The new web application significantly reduced the number of screens and simplified navigation. A responsive layout was implemented to make web pages render on a variety of devices and screen sizes. The new system also mirrors the manual paper case filing system with an automated paperless case workflow with the ability to track the status of any case file.

Section 4. Technical Design Guidelines

This section provides an overview of the architectural goals and describes the guiding design principles taken into consideration while designing and developing NuTRIS-Web solution.

4.1 Goals

The overall architectural goal of the system is to provide a highly available, scalable NuTRIS-Web interface by utilizing industry best practices including development guidelines, design patterns, and the latest .NET Core framework. Additional design goals include

- **Scalability:**

  Scalability is the application's ability to scale both up and down to support varying numbers of users or transaction volumes. The application design should support n-tier environment. The application should be able to scale horizontally (by adding more servers) or vertically (by increasing hardware capacity or software efficiency).

- **Flexibility:**

  Flexibility is the application's ability to accommodate new requirements without affecting the existing operations. This relies on a modular architecture, which isolates the complexity of integrating presentation and business logic from each other in order to allow for the easy integration of new technologies and processes within the application.

- **Manageability:**

  Designing an application for manageability involves providing an information infrastructure that allows the application and all its important supporting services and devices to be monitored for possible corrective and preventive action.

- **Extensibility:**

  Extensibility is the capacity to add new functionality to the development environment. It is also the ability to dynamically introduce new functionality. Extensibility can be achieved by using concepts, such as inversion of control and dependency injection patterns.

- **Availability:**

  Availability is the ability of the system to be in a state to perform a required operation at a given instance of time assuming that the external resources are provided. Using layered architecture and developing high-quality code in addition to fail-safe infrastructure design will help ensure high availability.

4.2 Principles

Guiding Principles

The NuTRIS-Web architecture is designed according to the guiding principles in Table 1.

Table 1: Guiding Principles

No.	Principle	Description
1	Design and Develop Application Software Components for Reusability while promoting continuous improvement of processes	This principle emphasizes two main characteristics of open systems standards: designing application software as components of an overall system and designing components for reusability. Together, these concepts constitute the minimum requirements for designing and deploying adaptable IT solutions that can evolve with the business needs.
2	Design and develop application using the Clean architecture model	This principle emphasizes on partitions the concerns of the application into stacked groups (layers). Layered architecture focuses on the grouping of related functionality within an application into distinct layers that are stacked vertically on top of each other. Functionality within each layer is related by a common role or responsibility.
3	Utilizing the pluggable Kendo UI products	This principle emphasizes on using the pluggable Kendo UI products to reduce the development time in a cost-effective manner.
4	Promote the Use of Web based Technology	The Internet and its related web-based technologies are the most significant advancements in information systems to occur in the last decade. The Internet, intranets and extranets offer new channels for enhanced communications directly between customers and suppliers. As web-based technologies continue to rapidly evolve, they are setting new standards and changing paradigms for using computers and networks to solve business problems.

Design Principles

The NuTRIS-Web architecture follows design principles described in Table 2.

Table 2: Design Principles

No.	Principle	Description
1	Separation of Concerns	Application has to be divided into distinct features with as little overlap in functionality as possible. The important factor is minimization of interaction points to achieve high cohesion and low coupling. However, separating functionality at the wrong boundaries can result in high coupling and complexity between features even though the contained functionality within a feature does not significantly overlap.
2	Persistence Ignorance (PI)	PI allows business entities to be persisted without the underlying knowledge of the caching technology. Types that implements PI in .NET can be referred as Plain Old CLR Objects or POCOs.
3	Dependency Inversion (DI)	Modules within the application should depend on abstraction rather than concrete implementation. This can be achieved by using a DI framework. DI is key when developing loosely coupled applications.
4	Single Responsibility Principle	Each component or module should be responsible for only a specific feature or functionality, or aggregation of cohesive functionality.
5	Do not repeat yourself (DRY)	Intent has to be specified in one place. For example, in terms of application design, specific functionality should be implemented in only one component; the functionality should not be duplicated in any other component.
6	Test-Driven Development (TDD)	In TDD, tests will be written before the code has been developed for a given requirement. This approach reduces bugs in the system and makes refactoring of code easier.
7	Well-Defined Interfaces	A component should not know the internal workings of another component and should only communicate via an agreed upon interface or contract. This helps ensure that the solution is easy to maintain and extend.
8	Bounded Context	In Domain Driven Design, Bounded Context defines a clear boundary for each business model. By using repository and factory patterns, interactions with the domain models can only happen within this boundary.

Section 5. Architectural Design

This section outlines the system and hardware architecture design of NuTRIS-Web.

5.1 Assumptions and Constraints

The following tables list the architectural assumptions and constraints for NuTRIS-Web.

Table 3: Architectural Design - Assumptions

Assumptions	Comments
.NET Core 2.2 framework will be used on the project.	
Entity Framework Core will be utilized on the project.	Entity Framework Core is offered as a NuGet package from Microsoft.
NTPR staff will use Google Chrome browser.	NuTRIS-Web system can be best viewed in Google Chrome browser.

Table 4: Architectural Design - Constraints

Constraints	Comments
Uploaded files cannot be viewed in browser.	Files uploaded to NuTRIS-Web should be downloaded before they can be viewed.

5.2 Hardware Architecture

NuTRIS-Web is hosted within DTRA’s network (UNET).

Web Server

NuTRIS-Web application is hosted on a Windows 2016 Core Server running Http.sys, which acts as a Web Server for the ASP .NET core application.

Database Server

SQL Server 2016 acts as the Database server where the data is stored. As the legacy NuTRIS PowerBuilder application’s data is stored in an older version of SQL Server and since it will be easier to migrate the data from the old system, SQL Server has been selected as the data store for NuTRIS-Web application.

File Server

Files uploaded to the NuTRIS-Web application are saved to the File Server which runs Windows 2016 Core.

UNET network users who are located at DTRA can connect to NuTRIS-Web application directly through a web browser using Windows authentication. External users should connect to UNET through Global Protect VPN using CAC for authentication; once connected, they will be able to access NuTRIS-Web same as internal users.

Section 6. Software Architecture

The NuTRIS-Web software architecture is designed to meet all the technical and operational requirements while optimizing common quality attributes, such as performance, security, and manageability. Building the system involves a series of decisions based on a wide range of factors. Each of these decisions can have considerable impact on the quality, performance, maintainability, and overall success of the application.

NuTRIS-Web employs Clean architecture as well as other methodologies such as component-based architecture and object-oriented architecture.

6.1 Clean Architecture

In Clean architecture, complexity of the application is separated into responsibilities and concerns. The architecture primarily consists of Dependency Inversion (DI) and Domain Driven Design (DDD).

Dependency Inversion (DI)

When developing a software solution, the business requirements are broken into smaller components. As the size of the solution increases so does the dependencies between the components. If the dependencies are concrete and rigid, maintaining and testing the components will be very difficult. The two main principles of DI are[R1]:

- High-level modules should not depend on low-level modules. Both should depend on abstractions (e.g., interfaces).
- Abstractions should not depend on details. Details (concrete implementations) should depend on abstractions.

Domain Driven Design (DDD)

DDD is a software philosophy where models are designed to closely mimic the underlying business processes and the relationship between them is explicitly defined. A Domain is a business area within which entities and functions are clearly defined. Bounded context defines a tangible boundary around the domains.

Clean architecture focuses on the grouping of related functionality within NuTRIS-Web into distinct layers that are stacked vertically. Layering the application appropriately helps to support a strong separation of concerns, which, in turn, supports flexibility and maintainability. The layers of the application may reside on the same physical computer (the same tier) or may be distributed over separate computers (n-tier), and the components in each layer communicate with components in other layers through well-defined interfaces/classes.

NuTRIS-Web architecture is divided into the following layers:

- Presentation or UI Layer
- Application Core Layer
- Infrastructure Layer
- Data Layer

Each Layer in a Clean architecture solutions has its responsibility clearly defined.

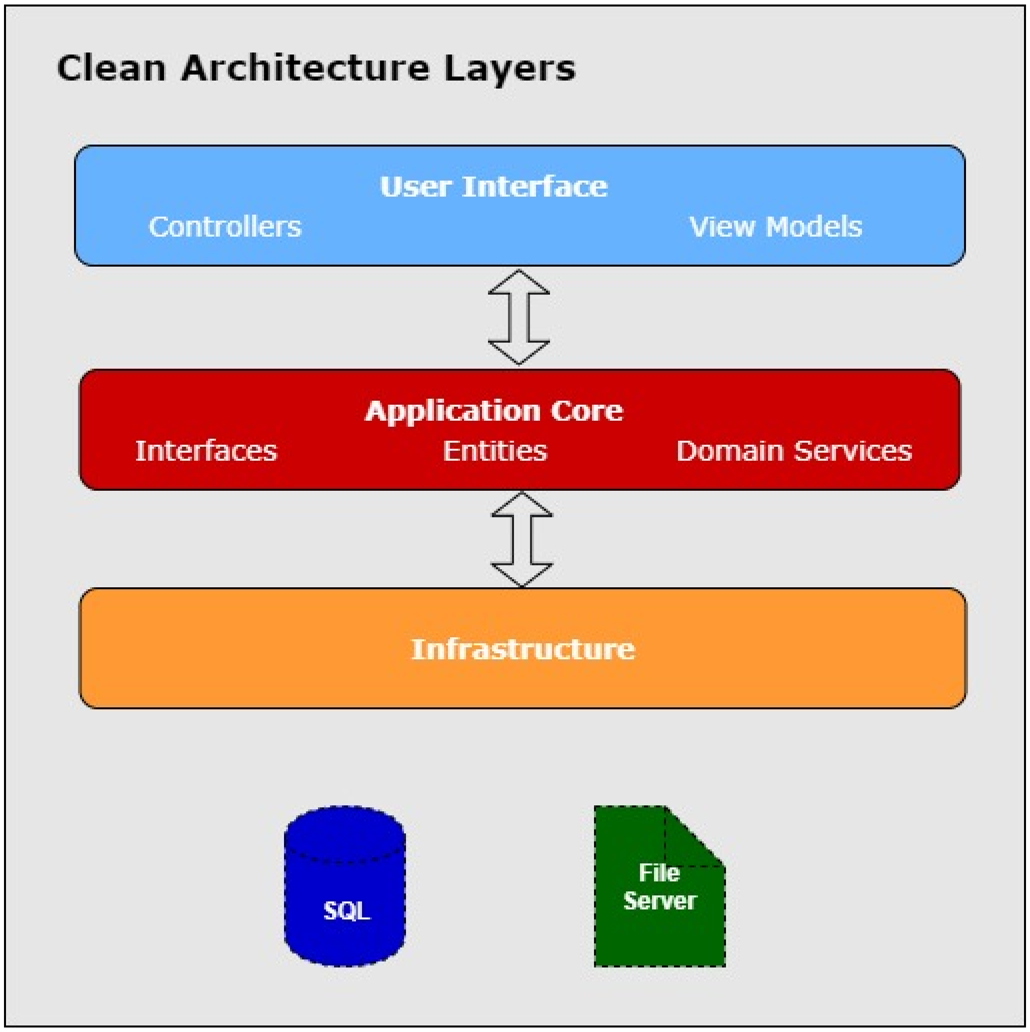


Figure 1: Clean Architecture

6.2 Layers

Presentation Layer

The presentation layer contains the components that implement and display the user interface and manage user interaction.

NuTRIS-Web presentation layer contains the following components:

- **User interface components:**

  User interface components are a set of components that manage the display of information and allows the user to interact with the system. User interface components will be implemented in NuTRIS-Web .NET Core Razor Pages and Vue.js Pages. These components are generally referred to as "Views" and perform the operations of acquiring and displaying the data to the user.

  These components are responsible for:
  - Using appropriate controls and validations for data input.
  - Managing visual layouts, styles, and the general appearance and navigation of the application.
  - Formatting, organizing data, and displaying it in useful visual styles.

- **User interface process components:**

  These components orchestrate the user interface elements and control user interaction. Users do not see user interface process components; however, these components perform a vital supportive role to user interface components. These components are generally referred to as "Application controllers" and manage the control flow in a given execution of a use case.
  - Managing control flow through the user interface components involved in a use case.
  - Encapsulating how exceptions affect the user process flow.
  - Maintaining internal business-related state, usually holding on to one or more business entities that are affected by the user interaction.

Application Core

The Application Core holds the business model, which includes entities, repositories, interfaces, and services. The Application Core layer is responsible for processing the application business logic. In NuTRIS-Web, the business layer is represented as a class library. The business logic is defined as any application logic that is concerned with the retrieval, processing, transformation, and management of application data; application of business rules and policies; and ensuring data consistency and validity.

NuTRIS-Web business layer contains the following components:

- **Domain Entities and Repositories:**

  Domain entities store data values and expose them through properties; they contain and manage business data used by an application and provide stateful programmatic access to the business data. Business entity components will be implemented in NuTRIS-Web as standalone classes (referred to as a "Class Library") and are responsible for processing the application logic and maintaining the application data. Business entity components are responsible for:

- Sequencing calls to other business logic or data access logic.
- Initiating and controlling business logic transactions.

- **Business Logic:**

  Business logic comprises the business rules pertaining to a business entity, for example calculation or conditional logic.

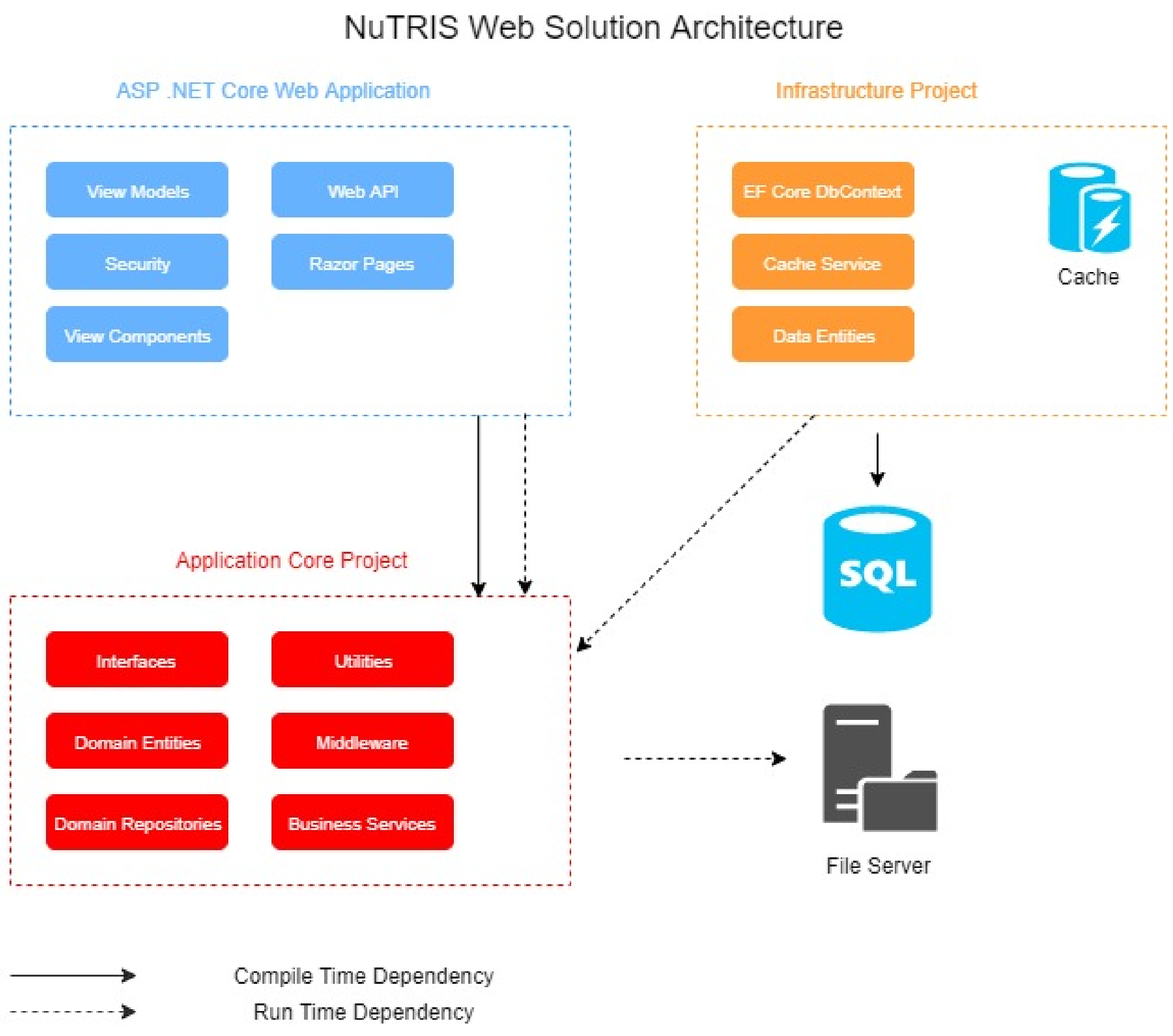


Figure 2: Solution Architecture

Infrastructure Layer

The Infrastructure Layer will typically include data access implementations. In a typical ASP.NET Core web application, this will include the Entity Framework (EF) DbContext, any

EF Core Migrations that have been defined, and data access implementation classes. The most common way to abstract data access implementation code is with the Repository design pattern. Repository design pattern provides a clean separation between the database objects and the domain layer. The Repository encapsulates the data objects and the CRUD operation performed over them via interfaces.

Data Layer

The data layer hosts the data access components to access the underlying data stores. The data access components centralize the common data access functionality in order to make the application easier to configure and maintain. The Platform framework will provide the data layer for NUTRIS-Web.

Section 7. Security Architecture

7.1 Authentication

Authentication is the process of verifying the identity of the system user typically by username and password. Since NuTRIS-Web is an intranet application and all its users are Windows users, the system authenticates using the Windows user name.

7.2 Authorization

After successful authentication of the system user by the web server, authorization determines access to the web resources for the authenticated user. This is achieved by role-based authorization where user roles and the actions they can perform on each web page are clearly defined. Authentication and Authorization can be managed through the Administration portal in NuTRIS-Web.

7.3 Encryption

ASP .NET Core framework provide APIs to secure sensitive data during transmission. URLs containing sensitive data are encrypted using .NET Core framework libraries. Sensitive data at rest will be encrypted using database encryption.

Section 8. Data Architecture

The NuTRIS-Web data architecture defines how the data is stored, managed, and used in the system. It establishes common guidelines for data operations that make it possible to predict, model, gauge, and control the flow of data in NuTRIS-Web.

NuTRIS-Web Database

The NuTRIS-Web database is designed for the day-to-day operational needs of the business and optimized for online transaction processing (OLTP).

Section 9. System Design

This section describes the concrete implementation of the architecture design at the component and the object level.

9.1 Design Considerations

The following design considerations were implemented in the NuTRIS-Web architecture.

Table 5: Design Considerations

No.	Consideration	Rational
1	Implementation of the Presentation layer using the user interface components and process components.	The NuTRIS-Web presentation layer will be implemented using the interface components and processes provided by .NET Core. These components will make the UI layer highly maintainable and easily testable.
2	Implementation of the business layer using business logic and entities.	The NuTRIS-Web business layer will be implemented using the business logic and entities. Business entities provide abstraction of the business rules and facilitate good design.

9.2 Exception Handling

Exception Handling is the process of gracefully handling runtime errors by providing meaningful notifications for the end users and logging the errors for later review by technical staff. NuTRIS-Web records diagnostic logs from the system to a SQL Server database using Serilog.

9.3 Hosting

ASP .NET Core offers multiple hosting options on different operating systems. In a Windows environment the following are the web servers available:

- Kestrel
- IIS
- Http.Sys

NuTRIS-Web uses Http.Sys as the web server on a Windows server. It supports Windows authentication; Kestrel doesn’t. Since reverse proxy is not needed, Http.sys has been chosen over IIS. HTTP.sys is production ready technology and runs in its own process. It protects against many types of attacks and is robust, secure and scalable

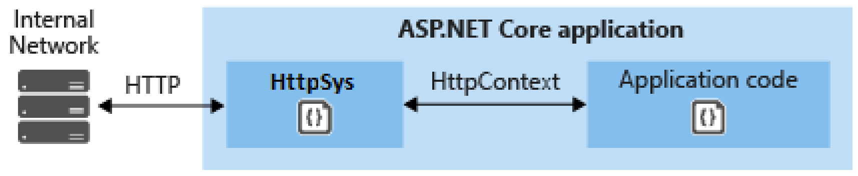


Figure 3: Http.Sys Server [R2]

9.4 SSL

Communication with NuTRIS-Web will be secured using HTTPS protocol. HTTPS encrypts the data between the browser and the web server by using SSL.

Section 10. Solution Structure

NuTRIS-Web solution consists of three projects separated by their respective architectural responsibility. Even though the solution has three projects, it is deployed as a single web site through which users will interact.

Infrastructure Project

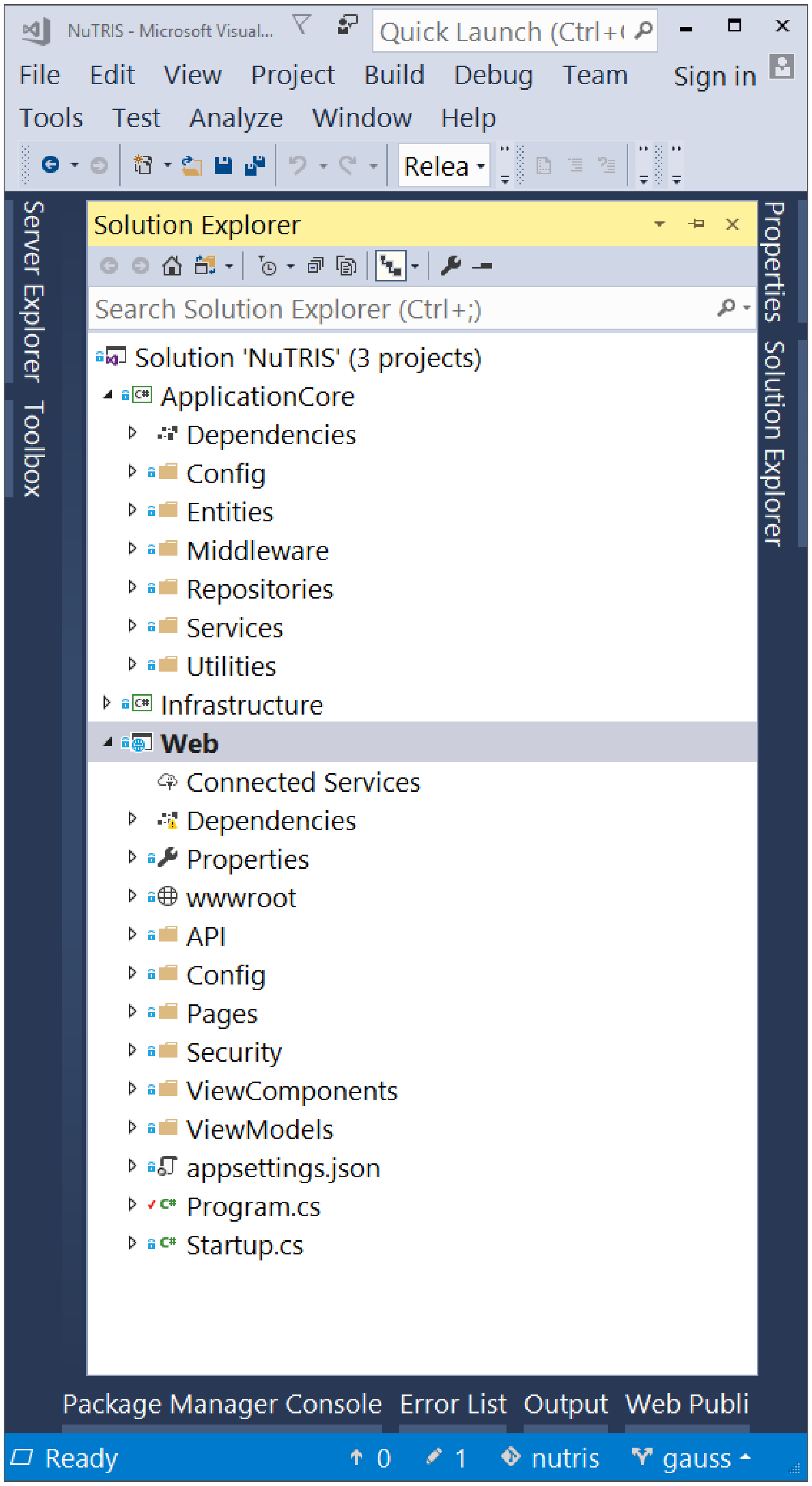
The infrastructure project consists of data access logic with POCO classes generated by Entity Framework. Manual code changes are never permitted in this project as they will be overwritten when new code is generated using EF commands to reflect changes in the data model.

Application Core Project

Domain layer which is part of the application core project consists of entities and repositories with CRUD operations defined; service layer classes defines the core business logic. Data transformation of objects between the layers is achieved using AutoMapper. Utilities, which has classes containing common functionality, and Middleware classes used in ASP .NET Core are part of the application core project.

Web Project

Web project primarily consists of UI elements – HTML Pages, Razor Pages, JavaScript files, CSS files, images, Kendo UI framework files. The security module which is responsible for the authentication and authorization of the web site resides in this project. View Models which are an abstraction of UI elements are stored in a separate folder. The configuration files hold information such as connection strings, file server paths etc. Dependency injection services are defined in the startup class; the web project acts as the entry point for the NuTRIS-Web application.


NuTRIS - Microsoft Visual...
Quick Launch (Ctrl+(
File Edit View Project Build Debug Team Sign in
Tools Test Analyze Window Help
Relea
Server Explorer
Toolbox
Properties
Solution Explorer
Search Solution Explorer (Ctrl+;)
Package Manager Console Error List Output Web Publi
Ready 0 1 nutris gauss

```
Solution 'NuTRIS' (3 projects)
  ApplicationCore
    Dependencies
    Config
    Entities
    Middleware
    Repositories
    Services
    Utilities
  Infrastructure
  Web
    Connected Services
    Dependencies
    Properties
    wwwroot
    API
    Config
    Pages
    Security
    ViewComponents
    ViewModels
    appsettings.json
    Program.cs
    Startup.cs
```

Figure 4: Solution Structure

Section 11. Data Model

11.1 Person Data Model

Figure 5 represents the Person data model and its direct table relationships. The tables capture general personnel information (i.e., address, phone, SSN, etc.) about the veteran.

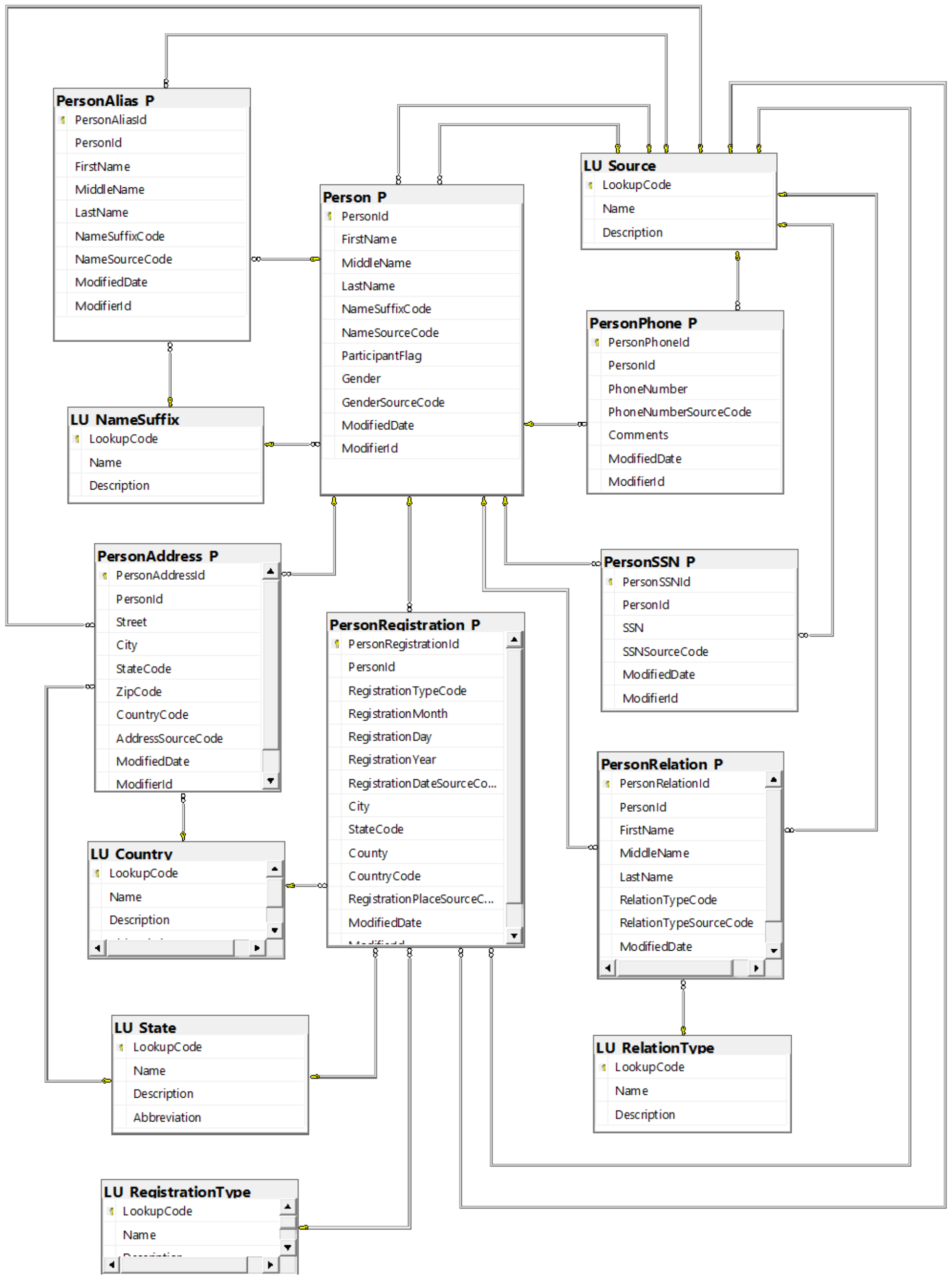


Figure 5: Person Data Model

11.2 Dose Data Model

Figure 6 represents the Dose data model and its direct table relationships. The tables contain information about the level of radiation the veteran was exposed to both internally and externally.

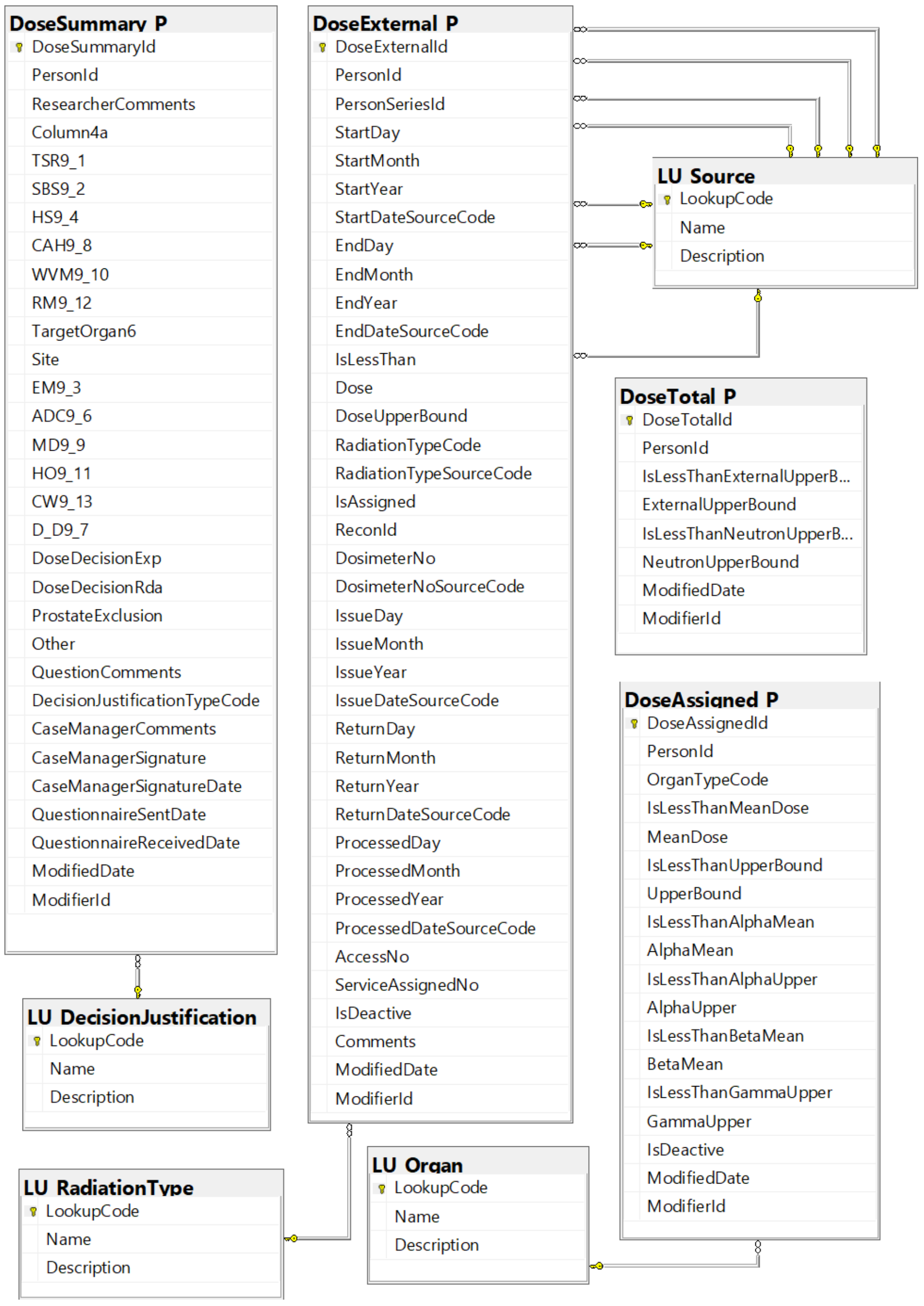


Figure 6: Dose Data Model

11.3 Case Data Model

Figure 7 represents the Case data model and its direct table relationships. The tables contain general information about the veteran’s case. In NuTRIS-Web, this information is entered when a new case is created for a veteran.

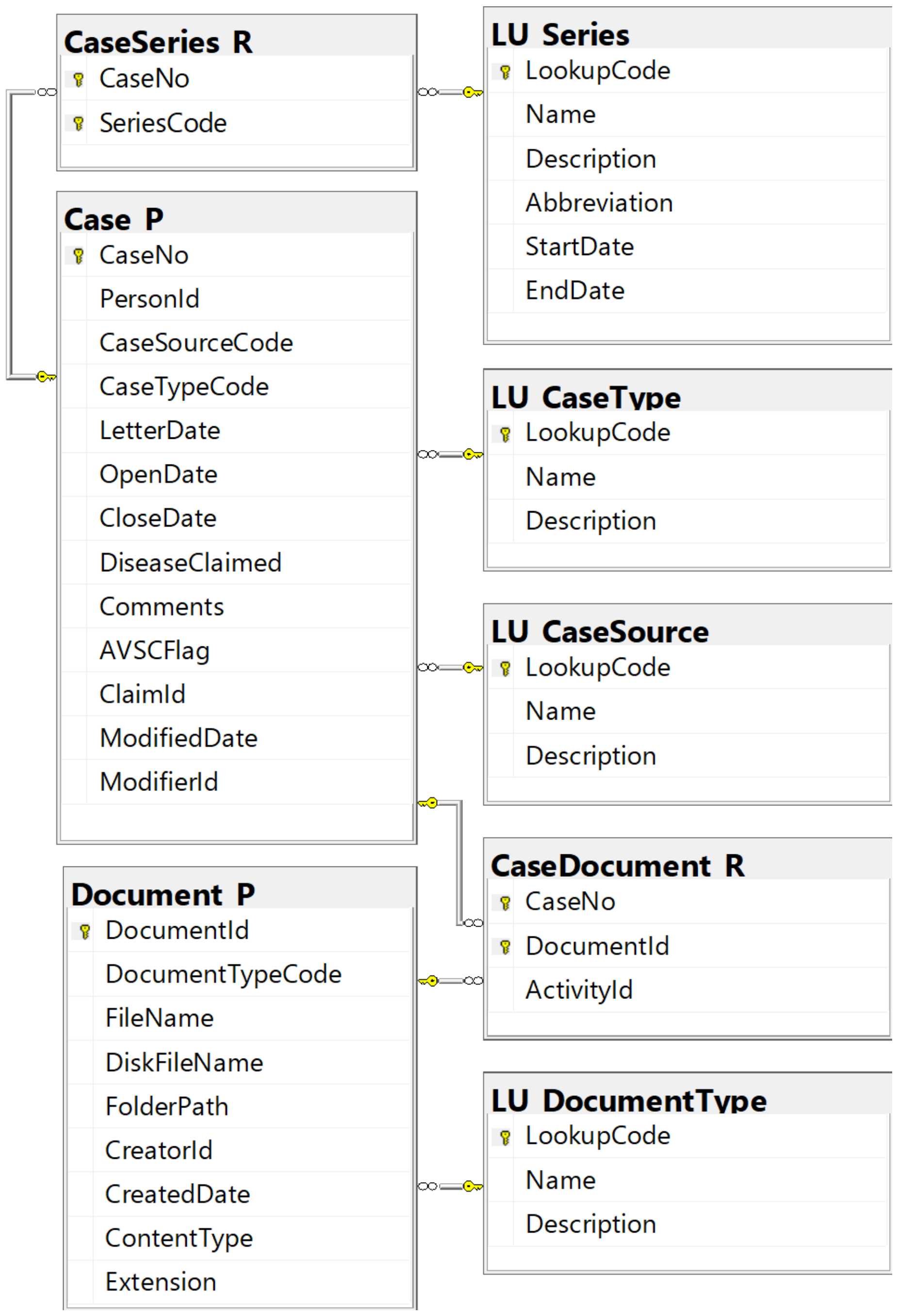


Figure 7: Case Data Model

11.4 Service Data Model

Figure 8 represents the Service data model and its direct table relationships. This table captures the service, series, and unit information for the veteran.

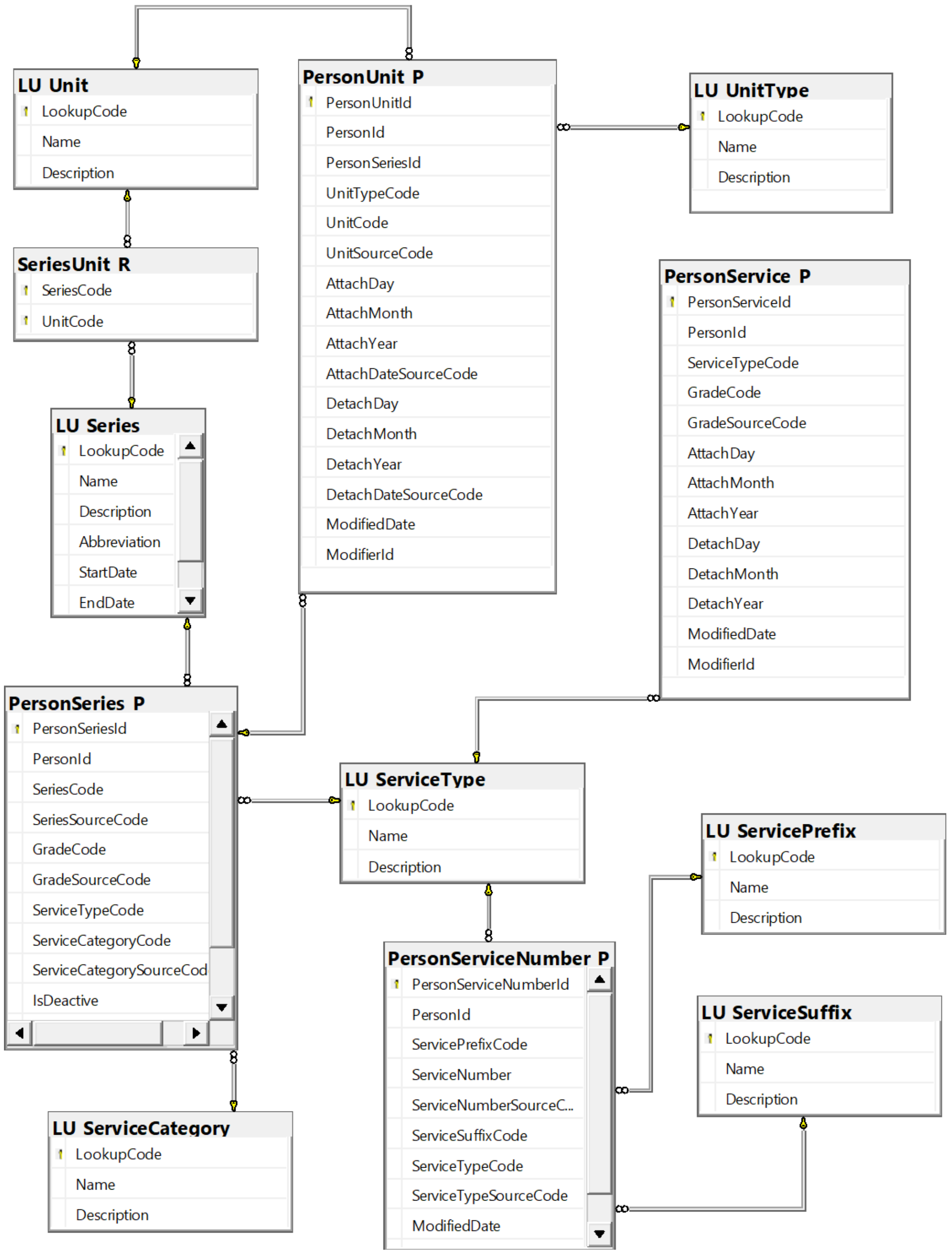


Figure 8: Service Data Model

11.5 Workflow Data Model

Figure 9 represents the Workflow data model and its direct table relationships. The workflow tracks case actions within NuTRIS-Web through workflow phase transitioning, case activities, and tasks.

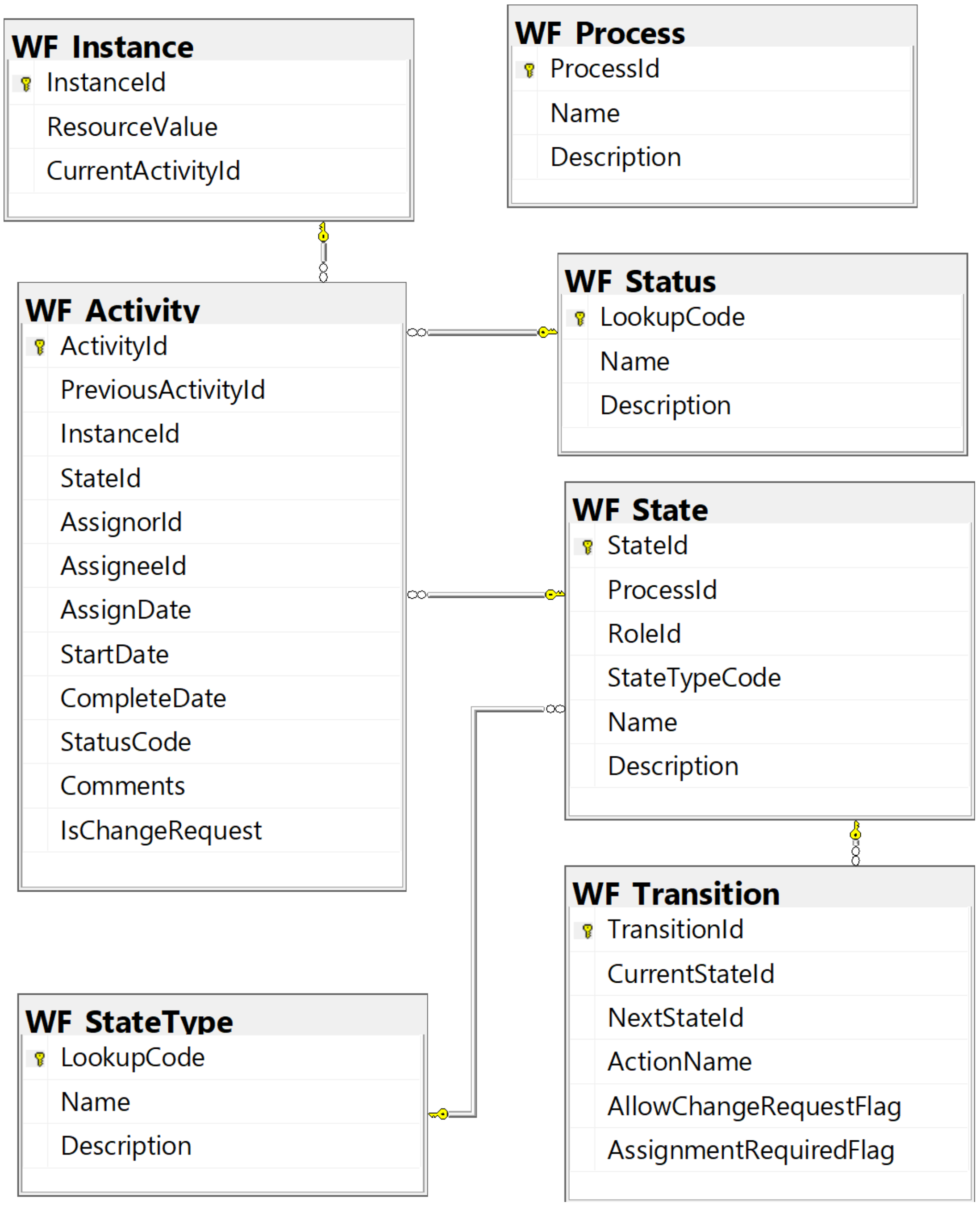


Figure 9: Workflow Data Model

11.6 Legacy Data Model

Figure 10 represents the Legacy data model and its direct table relationships. The tables hold data from the legacy NuTRIS PowerBuilder application and contains historical case information for the veteran.

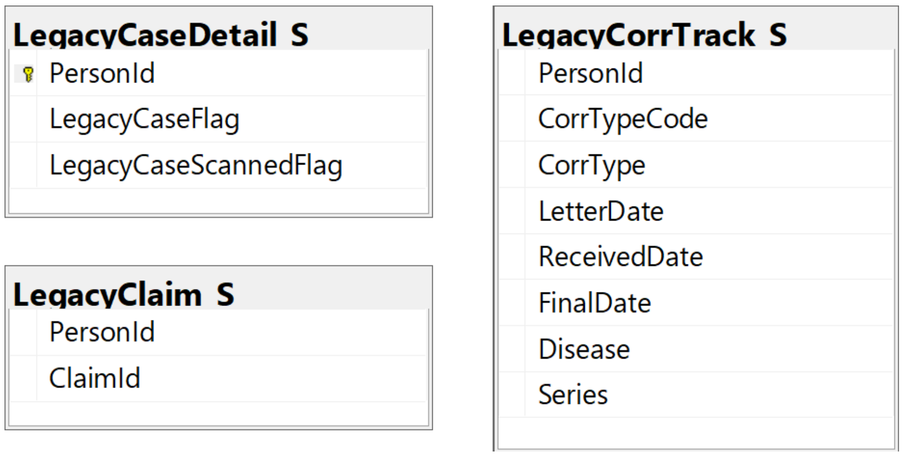


Figure 10: Legacy Data Model

11.7 User Data Model

Figure 11 represents the User data model which contains user role assignments. Access to various functions on NuTRIS-Web are controlled by the roles.

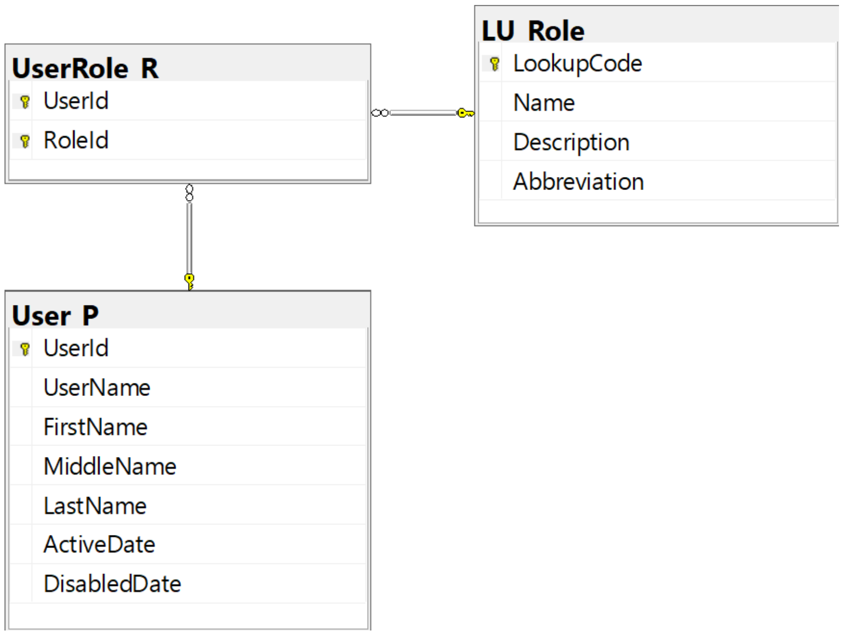


Figure 11: User Data Model

Section 12. Abbreviations, Acronyms and Symbols

Table 6: Acronyms List

Term	Definition
API	Application Programming Interface
CLR	Common Language Runtime
DDD	Domain Driven Design
DI	Dependency Inversion
DRY	Do not Repeat Yourself
EF	Entity Framework
HTTPS	Hyper Text Transfer Protocol Secure
MVC	Model View Controller
NTPR	Nuclear Test Personnel Review
NuTRIS	Nuclear Test Research Information System
OLTP	Online Transactional Processing
PI	Persistence Ignorance
POCO	Plain Old CLR Object
SOA	Service-Oriented Architecture
SQL	Structured Query Language
SSL	Secure Sockets Layer
UI	User Interface
URL	Uniform Resource Locator

Section 13. References

Table 7: References

Ref #	URL
[R1]	https://en.wikipedia.org/wiki/Dependency_inversion_principle
[R2]	https://docs.microsoft.com/en-us/aspnet/core/fundamentals/servers

Appendix A. User Interface Design

NuTRIS-Web user interface is designed and developed according to the following standards:

1. All NuTRIS-Web functional modules, processes, and features currently in production shall be captured in detailed requirements that can be traced to the new user interface features. These requirements are created as stories in the agile process and maintained in JIRA.
2. Web pages will meet the following guidelines:
   - Proper markup to identify the page title (i.e. an <h1> element)
   - Meaningful titles on all pages
   - Providing alternate text equivalents for images and image map hot spots
   - Providing null text equivalents on decorative images
   - Identifying row and column headers for data tables
   - Programmatically associating labels with form fields
   - Providing indicator to show important information or required fields
   - Alerting users with timed responses
   - Supporting browser settings for enlarging text and user style sheets
   - Using consistent navigation mechanisms and style of presentation throughout the site
   - Supporting keyboard navigation of web pages
   - Testing for compatibility with assistive technology in pages that use JavaScript for essential functions
   - Ensuring accessibility of non-HTML content such as PDF files, Microsoft Word documents, and PowerPoint presentations

Appendix B. Engineering Guidelines

NuTRIS-Web will use the standard engineering guidelines provided by Microsoft:
https://github.com/aspnet/AspNetCore/wiki/Engineering-guidelines